UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/632,569	08/01/2003	Douglas Earl Hays	RSW920030147US1	5013

36736 7590 06/26/2006

DUKE W. YEE
YEE & ASSOCIATES, P.C.
P.O. BOX 802333
DALLAS, TX 75380

EXAMINER	
STEELMAN, MARY J	

ART UNIT	PAPER NUMBER
2191	

DATE MAILED: 06/26/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/632,569	HAYS ET AL.	
	Examiner	Art Unit	
	Mary J. Steelman	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *01 August 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-35* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☐ Claim(s) *1-35* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *01 August 2003* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-35 are pending.

Drawings

2. FIG. 4, #400 is missing in the drawing.

Specification

3. The use of the trademark JAVA has been noted in this application. It should be capitalized wherever it appears and be accompanied by the generic terminology.

Although the use of trademarks is permissible in patent applications, the proprietary nature of the marks should be respected and every effort made to prevent their use in any manner which might adversely affect their validity as trademarks.

Claim Rejections - 35 USC § 101

4. 35U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 1-4 and 7-8 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Claim 1 is directed towards "identifying", "parsing" and "ascertaining", but do not produce a practical application established by a useful, concrete, tangible result and, thus, are non-statutory. (Claims 5 & 6 are cured by prepending a next segment to the proposed package name, thus creating a concrete, tangible result.)

Claims 12-15 & 18- 19 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Claim limitations do not produce a practical application established by a useful, concrete, tangible result and, thus, are non-statutory.

Claims 23-26, 30, and 34 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Claim limitations do not produce a practical application established by a useful, concrete, tangible result and, thus, are non-statutory.

Claims 23-33 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. It appears to Examiner that claim limitations, "a computer program product in a computer readable medium", as defined in the Specification (page 19, lines 13-26), are meant to include non-statutory forms analogous to such examples as signals and carrier waves.

Claim Rejections - 35 USC § 112

5. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

6. Claims 7, 8, 11, 16, 19, 22, 27, 28, 30, and 33 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

See MPEP 7.35.01 Trademark or Trade Name as a Limitation in the Claim

Claims 7, 8, 11, 16, 19, 22, 27, 28, 30, and 33 contain the trademark/trade name JAVA. Where a trademark or trade name is used in a claim as a limitation to identify or describe a particular

material or product, the claim does not comply with the requirements of 35 U.S.C. 112, second paragraph. See Ex parte Simpson, 218 USPQ 1020 (Bd. App. 1982). The claim scope is uncertain since the trademark or trade name cannot be used properly to identify any particular material or product. A trademark or trade name is used to identify a source of goods, and not the goods themselves. Thus, a trademark or trade name does not identify or describe the goods associated with the trademark or trade name. In the present case, the trademark/trade name is used to identify/describe byte code programming language and, accordingly, the identification/description is indefinite.

The trademark JAVA is improperly relied upon in the claims to incorporate the technical features of a particular programming language environment. However, the trademark JAVA can only properly define the source of the programming language environment, namely Sun Microsystems, Inc. Accordingly, the identification/description is indefinite.

Sun Microsystems, Inc. is the sole producer and/or licenser of JAVA products. The trademark JAVA identifies the source of the products and not the products themselves. In contrast, for example, C++ is a name used in trade to identify a particular nonproprietary programming language conforming to an accepted standard. Products and services incorporating the name C++ are produced by numerous sources. Further, the technologies identified using the trademark JAVA are continuously evolving. An example of this evolution can be found in "JSR 14: Add Generic Types To The Java™ Programming Language", which describes a proposed amendment to the JAVA Language Specification submitted by Sun Microsystems, Inc., in 1999 and pending

approval by the JAVA COMMUNITY PROCESS Program. In view of the statements presented above, it is asserted that the trademark JAVA has no fixed definite technical meaning. Accordingly, a rejection under 35 U.S.C. 112, second paragraph, based on the use of the trademark JAVA as a limitation in a claim, is proper.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. Claims 1-35 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent Application Publication 20020188935A1 to Hertling et al., in view of "Docjar – Source code: org/jboss/mx/loading/ClassLoaderUtils.java" (hereinafter Docjar).

Per claim 1:

A process in a data processing system for identifying package names, the process comprising the computer implemented steps of: responsive to receiving a selection of a class file, identifying a path for a class file; parsing the path to identify a set of sequential segments; and ascertaining a package name for the class using the set of sequential segments, wherein the package name is ascertained without disassembling the class file.

Hertling: [0014-0015]-"The virtual machine typically uses a software component called a 'class loader' to locate a requested class file. The virtual machine relies upon a previously defined 'class path' to define the locations of directories and subdirectories where the class files available to the virtual machine are stored...the class path is an environment variable listing multiple file path entries, each path ending in a filename or directory. The class loader typically locates required class files by relying upon a standardized naming convention for the class files of the virtual machine...For example, for the JAVA virtual machine, class packages are named in lowercase with periods indicating directory levels. The prefix of the class package name is a top-level domain name, such as com, edu, gov, mil, net, org, or a two-letter country identifier. The subsequent components of the class package name vary according to the organization that developed the class file itself, but typically the components of the name following the domain may include a reference to the developer organization, a specific division, department, project, machine, or login name..."

Thus, Official Notice is taken that a directory name for a given class may be parsed, identifying the dot (period) delimiters to ascertain the package name. JAVA has defined two utilities useful for the parse / transformation. See examples shown in the Docjar document. At page 3 the getPackageName() utility is disclosed. At page 9 the toPackageName() utility is disclosed. The package name is ascertained by parsing the path and discarding the last segment (the class name). This is done without disassembling the class file.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hertling, to include the defined JAVA utilities to parse the directory path name in order to determine the package name, as disclosed in Docjar, because Hertling recognized the 'ClassNotFound' [0007] exception occurs frequently as developers update versions, and Hertling [0008] recognized the need to find required packages and class files for download.

Hertling [0026] "At its most efficient, distribution of new virtual machine applications may require nothing more than identifying the name of the main class file for a new virtual machine application. As the main class file itself specifies the additional class files required to execute the virtual machine application, merely attempting to execute the main class file would result in firs, accessing the class file repository for the main class file, then as the main class file was executed, each of the additional class files and class file packages required for the application would be downloaded and executed. An entire application could be distributed on a hugh scale, merely by publishing the name of that application's main class file."

Per claim 2:

-the class file is on a local file system.

Hertling: [0016]-"class files are located in the file system of the virtual machine"

Per claim 3:

-receiving a selection of the class file, wherein the selection includes information sufficient to

identifying the path for the class file.

Hertling: [0016]-"and their locations are specified in the class path."

Per claim 4:

-the parsing step includes: identifying segments in the set of sequential segments using delimiters in the path.

Hertling: [0015]-"class packages are named in lowercase with periods indicating directory levels."

Per claim 5:

-the ascertaining step includes: selecting first segment containing a base class name to form a proposed package name; submitting the proposed package name to a Java virtual machine; responsive to the proposed package name being an incorrect name, prepending a next segment to the proposed package name; and responsive to prepending the next segment, submitting the current package name to the Java virtual machine.

Hertling disclosed that JAVA class names are reflected by the class path. The class path is a period (dot) delimited name. At [0015], "Representative class package names include edu.abc.csjohns.banana, or com.ibm.eng-static.v3, or com.acme.server.servlet.extension. Thus it may be seen that using defined utilities to parse the directory string (and strip off the class name), a class name may be used to determine the package name. (Prepending each segment to the left.)

Per claim 6:

-the first segment is selected as being a first segment on a right side of the set of sequential segments.

Hertling: [0015] As noted above, using the segments, as determined by the dot delimiter, start with the first segment to the right to create the package name. See the results of the above noted utilities.

Per claim 7:

-the process is located in a Java class loader.

Hertling: [0014]-JAVA class loader

Per claim 8:

-the class file is a Java class file.

Hertling: [0015]-JAVA class file

Per claim 9:

A process in a data processing system for identifying a package name for a class file, the process comprising the computer implemented steps of: receiving a selection of a class file; identifying a path for the class file using the selection; parsing the path to form an ordered set of segments; selecting an unselected segment from the ordered set of segments; adding the unselected segment to a set of selected segments; generating a proposed package name using the set of selected segments; submitting the proposed package name to a virtual machine for loading; and repeating

the selecting, adding, generating, and submitting steps in response to the proposed package name being an incorrect package name, wherein the package name is identified without examining code in the class file.

Hertling: See rejection of claim 1 above. Should the generated proposed name be incorrect, the error message will be noted [0007]. The routine iterates to find the package name.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hertling, to include the defined JAVA utilities to parse the directory path name in order to determine the package name, as disclosed in Docjar, because Hertling recognized the 'ClassNotFound' [0007] exception occurs frequently as developers update versions, and Hertling [0008] recognized the need to find required packages and class files for download.

Per claim 10:

-wherein the code is a set of bytecodes.

Official Notice is taken that the compiled JAVA code to be run in a JAVA virtual machine is bytecodes. Class files are compiled JAVA bytecode.

Per claim 11:

-the virtual machine is a Java virtual machine.

Hertling: [0015] –JAVA virtual machine

Per claim 12:

A data processing system for identifying package names, the data processing system comprising: identifying means responsive to receiving means for receiving a selection of a class file, for identifying a path for a class file; parsing means for parsing the path to identify a set of sequential segments; and ascertaining means for ascertaining a package name for the class using the set of sequential segments, wherein the package name is ascertained without disassembling the class file.

See rejection of limitations as addressed above.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hertling, to include the defined JAVA utilities to parse the directory path name in order to determine the package name, as disclosed in Docjar, because Hertling recognized the 'ClassNotFound' [0007] exception occurs frequently as developers update versions, and Hertling [0008] recognized the need to find required packages and class files for download.

Per claim 13:

-the class file is on a local file system.

See rejection of limitations addressed in claim 2 above.

Per claim 14:

-receiving means for receiving a selection of the class file, wherein the selection includes information sufficient to identifying the path for the class file.

Hertling: [0015]-"The class loader typically locates required class files by relying upon a standardized naming convention."

Per claim 15:

-the identifying means is a first identifying means and wherein the parsing means includes: second identifying means for identifying segments in the set of sequential segments using delimiters in the path.

See rejection of limitations as addressed in claim 9 above16. The data processing system of claim 12, wherein the ascertaining means includes: selecting means for selecting first segment containing a base class name to form a proposed package name; first submitting means for submitting the proposed package name to a Java virtual machine; prepending means, responsive to the proposed package name being an incorrect name, for prepending a next segment to the proposed package name; and second submitting means responsive to prepending the next segment, for submitting the current package name to the Java virtual machine.

Per claim 16:

-selecting means for selecting first segment containing a base class name to form a proposed package name;

-first submitting means for submitting the proposed package name to a Java virtual manchin;

-prepending means, responsive to the proposed package name being an incorrect name, for prepending a next segment to the proposed package name;

-second submitting means responsive to prepending the next segment, for submitting the current package name to the Java virtual machine.

See rejection of limitations as addressed in claim 9 above.

Per claim 17:

-the first segment is selected as being a first segment on a right side of the set of sequential segments.

Official Notice is taken that the standardized naming convention for class files, shows the root directory at the left. A parsed path name starts the parse from the right. See utilities noted above to determine a package name.

Per claim 18:

-the process is located in a Java class loader.

Hertling: [0014]-JAVA class loader.

Per claim 19:

-the class file is a Java class file.

Hertling: [0015]-JAVA class file

Per claim 20:

A process in a data processing system for identifying a package name for a class file, the data processing system comprising: receiving means for receiving a selection of a class file;

identifying means for identifying a path for the class file using the selection; parsing means for parsing the path to form an ordered set of segments; selecting means for selecting an unselected segment from the ordered set of segments; adding means for adding the unselected segment to a set of selected segments; generating means for generating a proposed package name using the set of selected segments; submitting means for submitting the proposed package name to a virtual machine for loading; and repeating means for repeating initiation of the selecting means, adding means, generating means, and submitting means in response to the proposed package name being an incorrect package name, wherein a package name is identified without examining code in the class file.

See rejection of limitations as addressed in claims 1, 5, 9, 12 above.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hertling, to include the defined JAVA utilities to parse the directory path name in order to determine the package name, as disclosed in Docjar, because Hertling recognized the 'ClassNotFound' [0007] exception occurs frequently as developers update versions, and Hertling [0008] recognized the need to find required packages and class files for download.

Per claim 21:

-the code is a set of bytecodes.

Official Notice is taken that the compiled JAVA code to be run in a JAVA virtual machine is bytecodes. Class files are compiled JAVA bytecode.

Per claim 22:

-the virtual machine is a Java virtual machine.

Hertling: [0015] –JAVA virtual machine

Per clam 23:

A computer program product in a computer readable medium for identifying package names, the computer program product comprising: first instructions responsive to receiving a selection of a class file, for identifying a path for a class file; second instructions for parsing the path to identify a set of sequential segments; and third instructions for ascertaining a package name for the class using the set of sequential segments, wherein the package name is ascertained without disassembling the class file.

See rejection of limitations as addressed in claims 1-8 above. See [0027].

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hertling, to include the defined JAVA utilities to parse the directory path name in order to determine the package name, as disclosed in Docjar, because Hertling recognized the 'ClassNotFound' [0007] exception occurs frequently as developers update versions, and Hertling [0008] recognized the need to find required packages and class files for download.

Per claim 24:

-the class file is on a local file system.

Hertling: [0016]-"class files are located in the file system of the virtual machine"

Per claim 25:

-fourth instructions for receiving a selection of the class file, wherein the selection includes information sufficient to identifying the path for the class file.

Herlting: [0015]-"The class loader typically locates required class files by relying upon a standardized naming convention for the class files of the virtual machine.

Per claim 26:

-the second instructions includes: sub-instructions for identifying segments in the set of sequential segments using delimiters in the path.

Hertling disclosed that periods (identify segments using delimiters) indicating directory levels are used in package names. Using the above noted defined utilities, a package name may be determined.

Per claim 27:

-the third instructions includes: first sub-instructions for selecting first segment containing a base class name to form a proposed package name; second sub-instructions for submitting the proposed package name to a Java virtual machine; third sub-instructions for responsive to the proposed package name being an incorrect name, prepending a next segment to the proposed

package name; and fourth sub-instructions for responsive prepending the next segment submitting the current package name to the Java virtual machine.

See rejection of limitations addressed at least at claims 1, 5, 9, and 12 above.

Per claim 28:

-the first segment is selected as being a first segment on a right side of the set of sequential segments.

Hertling disclosed a standardized naming convention. The first segment on the right (of the period delimiter) is the most nested directory of the path and defines the first location to look for the class name from which to deduct the package name.

Per claim 29:

-the process is located in a Java class loader.

Hertling: [0014]-JAVA class loader.

Per claim 30:

-the class file is a Java class file.

Hertling: [0015]-JAVA class file

Per claim 31:

A computer program product in a computer readable medium for identifying a package name for a class file, the computer: first instructions for receiving a selection of a class file; second

instructions for identifying a path for the class file using the selection; third instructions for parsing the path to form an ordered set of segments; fourth instructions for selecting an unselected segment from the ordered set of segments; fifth instructions for adding the unselected segment to a set of selected segments; sixth instructions for generating a proposed package name using the set of selected segments; seventh instructions for submitting the proposed package name to a virtual machine for loading; and eighth instructions for repeating the initiation of fourth, fifth, sixth, and seventh instructions in response to the proposed package name being an incorrect package name, wherein a package name is identified without examining code in the class file.

This is a computer program product version of the above addressed limitations. See [0027].

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hertling, to include the defined JAVA utilities to parse the directory path name in order to determine the package name, as disclosed in Docjar, because Hertling recognized the 'ClassNotFound' [0007] exception occurs frequently as developers update versions, and Hertling [0008] recognized the need to find required packages and class files for download.

Per claim 32:

-the code is a set of bytecodes.

Official Notice is taken that the compiled JAVA code to be run in a JAVA virtual machine is bytecodes. Class files are compiled JAVA bytecode.

Per claim 33:

-the virtual machine is a Java virtual machine.

Hertling: [0015] –JAVA virtual machine

Per claim 34:

A data processing system comprising: a bus system; a memory connected to the bus system, wherein the memory includes a set of instructions; and a processing unit connected to the bus system, wherein the processing unit executes a set of instructions to identify a path for a class file in response to receiving a selection of the class file; to parse the path to identify a set of sequential segments; and to ascertain a package name for the class using the set of sequential segments, wherein the package name is ascertained without disassembling the class file.
See rejection of limitations as addressed in claim 1 above. Hertling disclosed a data processing system at [0013] and FIGs. 1 & 2.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hertling, to include the defined JAVA utilities to parse the directory path name in order to determine the package name, as disclosed in Docjar, because Hertling recognized the 'ClassNotFound' [0007] exception occurs frequently as developers update versions, and Hertling [0008] recognized the need to find required packages and class files for download.

Per claim 35:

A data processing system comprising: a bus system; a memory connected to the bus system, wherein the memory includes a set of instructions; and a processing unit connected to the bus system, wherein the processing unit executes a set of instructions to receive a selection of a class file; identify a path for the class file using the selection; parse the path to form an ordered set of segments; select an unselected segment from the ordered set of segments; add the unselected segment to a set of selected segments; generate a proposed package name using the set of selected segments; submit the proposed package name to a virtual machine for loading; and repeat instructions to select, add, generate, and submit in response to the proposed package name being an incorrect package name, wherein the package name is identified without examining code in the class file.

See rejection of limitations as addressed in claims 1-9 above. Hertling disclosed a data processing system at [0013] and FIGs. 1 & 2.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hertling, to include the defined JAVA utilities to parse the directory path name in order to determine the package name, as disclosed in Docjar, because Hertling recognized the 'ClassNotFound' [0007] exception occurs frequently as developers update versions, and Hertling [0008] recognized the need to find required packages and class files for download.

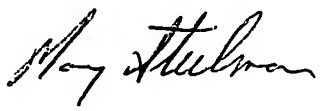
Conclusion

9. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached at (571) 272-3708. The fax phone number for the organization where this application or proceeding is assigned: 571-273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman 

06/20/2006